**Unit 3 FUNCTONS**

**Q1. Write a program to check whether a given number is even or odd using the functions.**

#include <stdio.h>

int checkOddEven(int n)

{

return (n & 1);

}

int main()

{

int n;

scanf("%d", &n);

printf("%d\n",n);

if(checkOddEven(n))

{

printf("The entered number is odd.");

}

else

{

printf("The entered number is even.");

}

**Footer**

return 0;

}

**Q2. Write a program to obtain two numbers and add them using functions.**

#include <stdio.h>

int add(int x,int y)

{

return x+y;

}

**Footer**

int main()

{

int a,b,c;

scanf("%d %d",&a,&b);

c=add(a,b);

printf("%d",c);

return 0;

}

**NEON NUMBER:**

Given a number (num) we need to check whether it is a Neon Number ( i.e. a number where the sum of digits of the square of the number is equal to the number ) and return “true” or “false” accordingly.

**Example:**

**Input**: num = 9

**Output**: true

***Explanation:***square of 9 is 9 \* 9 = 81 , sum of digit of square is 8 + 1 = 9 (i.e equal to given number).

**Q1. Misha likes to check and print whether the given number is a neon number or not. Write a program with a function prototype int neon(int) which helps her to complete this task**.

Note: A neon number is a number where the sum of digits of the square of the number is equal to the number.

#include<stdio.h>

int neon(int num)

{

int square,sum\_of\_digits=0;

square = num\*num;

while(square !=0) {

sum\_of\_digits = sum\_of\_digits + (square%10);

square /= 10;

}

return (sum\_of\_digits == num);

}

int main()

{

int num,isneon=0;

scanf("%d",&num);

isneon = neon(num);

isneon ? printf("%d is a neon number",num) : printf("%d is not a neon number",num);

}

**Q2. Write a C program to compute the power of a given number using a user-defined function. The function should get the base and exponent values as inputs from the user and return the value of the power of the base raised to the exponent value.**

#include<stdio.h>

void calculate\_power(int,int);

int main()

{

int b,e;

scanf("%d",&b);

scanf("%d",&e);

calculate\_power(b,e);

}

void calculate\_power(int b,int e)

{

int power=1;

while(e>0)

{

power=power\*b;

e--;

}

printf("%d",power);

}

**Q1. Write a program to convert decimal numbers to binary using the functions.**

#include<stdio.h>

long toBin(int dno)

{

long bno=0,remainder,f=1;

while(dno != 0)

{

remainder = dno % 2;

bno = bno + remainder \* f;

f = f \* 10;

dno = dno / 2;

}

return bno;

**Footer**

}

int main()

{

long bno;

int dno;

scanf("%d",&dno);

bno = toBin(dno);

printf("%ld",bno);

return 0;

}

**Q2. Write a program to input the radius of the circle from the user and find the diameter, circumference, and area of the given circle using a function.**

#include <stdio.h>

#include <math.h>

double getDiameter(double radius)

{

return (2 \* radius);

}

double getCircumference(double radius)

{

return (2 \* M\_PI \* radius);

}

double getArea(double radius)

{

return (M\_PI \* radius \* radius);

}

**Footer**

int main()

{

float radius, dia, circ, area;

scanf("%f", &radius);

dia = getDiameter(radius);

circ = getCircumference(radius);

area = getArea(radius);

printf("Diameter of the circle = %.2f units\n", dia);

printf("Circumference of the circle = %.2f units\n", circ);

printf("Area of the circle = %.2f sq. units", area);

return 0;

}

**Q1. Write a program having function declaration - void Fibonacci(int n), which generates Fibonacci series up to given n terms and prints them using call by value.**

#include<stdio.h>

void fibonacciSeries(int n)

{

int a=0, b=1, c,i=0;

while (i<n)

{

printf("%d ", a);

c = a+b;

a = b;

b = c;

i++;

}

}

int main()

{

int n;

scanf("%d", &n);

fibonacciSeries(n);

return 0;

}

**Q2. Define and call a function reverse\_number() to reverse a number passed to it and return the reversed number. The main() function has to accept the number from the keyboard.**

#include <stdio.h>

int reverse\_number(int n);

int main()

{

int num;

scanf("%d", &num);

printf("%d",reverse\_number(num));

return 0;

}

int reverse\_number(int n)

{

int rem, rev=0;

while(n!=0)

{

rem=n%10;

n=n/10;

rev=rev\*10+rem;

}

return rev;

}

**Day 3**

1. Write a program to find the sum of the series 1!/1+2!/2+3!/3+4!/4+5!/5+....+n!/n using the function.

Input Format

The input consists of the n value.

Output Format

The output prints the sum.

#include <stdio.h>

int fact(int n)

{

int num=0,f=1;

while(num<=n-1)

{

f =f+f\*num;

num++;

}

return f;

}

int main()

{

int n,i,sum=0;

scanf("%d",&n);

for(i=1;i<=n;i++) {

sum= sum+ (fact(i)/i);

}

printf("%d",sum);

return 0;

}

1. Entrance Test

"Success Academy" has arranged for an entrance test for High School students from rural villages. Those successful students of the test will be awarded a scholarship for their IIT/JEE preparations at Success Academy. Sunil, the co-coordinator and founder of the academy has given one problem for the first stage of the test. The problem goes like this:

Given two integers x and n, find the number of ways to express x as the sum of n-th powers of unique natural numbers. It is given that 1 <= n <= 20.

Sunil himself has not computed the solution to the problem. Write a recursive function to help him find the answer for the same to evaluate the students.

**Function Specifications**:

Use the function return type and the argument type as:

int countWays(int,int)

This recursive function should return the number of ways to express x as the sum of n-th powers of unique natural numbers.

**Input Format**

The first line of input contains the integer x.

The second line contains the integer n.

**Output Format**

Output the number of ways to express x as the sum of n-th powers of unique natural numbers in a single line.

#include<stdio.h>

#include<math.h>

int count(int x,int n,int num)

{

int i;

i=x-pow(num,n);

if(i==0)

{

return 1;

}

if(i<0)

{

return 0;

}

else

return count(i,n,num+1)+count(x,n,num+1);

}

int countWays(int x,int n)

{

return count(x,n,1);

}

**Footer**

int main()

{

int x,n;

scanf("%d%d",&x,&n);

printf("%d",countWays(x,n));

return 0;

}

**Cod**

Q1. Define and call a function sum\_of\_digits() to find the sum of digits of a given number and return the result. The main() function has to accept the number from the keyboard.

**Input Format**

The input consists of a number.

**Output Format**

The output prints the sum of the digits of the number.

**Sample InputSample Output**

1234

Sum of digits=10

#include <stdio.h>

int sum\_of\_digits(int n);

int main()

{

int num;

scanf("%d", &num);

printf("Sum of digits=%d\n",sum\_of\_digits(num));

return 0;

}

int sum\_of\_digits(int n)

{

int rem, sum=0;

while(n!=0)

{

rem=n%10;

n=n/10;

sum=sum+rem;

}

return sum;

}

Q2. Write a program having function declaration void factorial() which calculates the factorial of a given number and prints the result using call by value.

**Input Format**

The input consists of one integer.

**Output Format**

The output consists of a factorial number.

Sample InputSample Output

5 120

#include <stdio.h>

unsigned int factorial(unsigned int n)

{

int res = 1, i;

for (i = 2; i <= n; i++)

res \*= i;

return res;

}

int main()

{

int num;

scanf("%d",&num);

printf( "%d",factorial(num));

return 0;

}

**Day 4**

Q1. Write a program to find the maximum element in an array using functions.

**Input Format**

The first line of the input consists of the value of n.

The next input is the array elements.

**Output Format**

The output prints the maximum element in an array.

#include<stdio.h>

int n;

int findMaxElem(int arr1[],int n)

{

int i=1,mxelem;

mxelem=arr1[0];

while(i < n)

{

if(arr1[i]>mxelem)

mxelem=arr1[i];

i++;

}

return mxelem;

}

**Footer**

int main()

{

int n,mxelem,i;

scanf("%d",&n);

int arr[n];

for(i=0;i<n;i++)

{

scanf("%d",&arr[i]);

}

mxelem=findMaxElem(arr,n);

printf("%d",mxelem);

return 0;

}

Q2. Write a program to get n elements in an array and get a number to be multiplied by the array elements using a function. Display the multiplied result in descending order.

**Input Format**

The first input consists of the value of n.

The next input is the array elements.

The third line consists of an integer

**Output Format**

The output prints the multiplied result in descending order.

#include<stdio.h>

void multiply(int arr[],int n,int mul);

void multiply(int arr[],int n,int mul)

{

int i;

for(i=0;i<n;i++) {

arr[i] \*= mul;

}

for(i=0;i<n;i++)

{

printf("%d ",arr[i]);

}

}

**Footer**

int main()

{

int n,i,mul,temp,j;

scanf("%d",&n);

int arr[n];

for(i=0;i<n;i++)

{

scanf("%d",&arr[i]);

}

scanf("%d",&mul);

multiply(arr,n,mul);

return 0;

}

**Cod**

1. Write a C Program to find the smallest element in An array using the Function

**Function Name:** find\_small

**Input Format**

The First Line Consists of the size of an Array

The next Line Consists of elements in an Array

**Output Format**

The Output Should display the Smallest Element in an Array

#include<stdio.h>

int find\_small(int [], int);

int main()

{

int n;

scanf("%d",&n);

int arr[n], small, i;

for(i=0; i<n; i++)

scanf("%d", &arr[i]);

small=find\_small(arr, n);

printf("The smallest element in the array is: %d",small);

return 0;

}

int find\_small(int arr[], int n)

{

int i=0, small;

small=arr[i];

while(i<n)

{

if(small>arr[i])

small = arr[i];

i++;

}

return small;

}

Q2. Write a program to test the function by reading a set of numbers, compute and display the sum of them using Call by Value using the function.

**Function name:sumofarray**

**Input Format**

The first line of the input consists of N, the size of the array.

The second line of input consists of array elements

**Output Format**

The output prints the sum of elements.

#include <stdio.h>

int sumofarray(int a[],int n)

{

int i,sum=0;

for(i=0; i<n; i++)

{

sum+=a[i];

}

return sum;

}

int main()

{

int i,n,sum;

scanf("%d", &n);

int a[n];

for(i=0; i<n; i++)

{

scanf("%d",&a[i]);

}

sum=sumofarray(a,n);

printf("%d",sum);

}

Day 6

Q1. Write a program to find the gcd of two numbers using recursion.

**Input Format**

The input consists of two numbers.

**Output Format**

The output prints the gcd of two numbers.

**Sample InputSample Output**

24 60

12

#include <stdio.h>

int gcd(int a, int b)

{

if(b == 0)

return a;

else

return gcd(b, a%b);

**Footer**

}

int main()

{

int num1, num2, hcf;

scanf("%d%d", &num1, &num2);

hcf = gcd(num1, num2);

printf("%d",hcf);

return 0;

}

Q2. Write a program to find the sum of array elements using recursion.

**Input Format**

The first line of the input consists of the value of n.

The next input is the array elements.

**Output Format**

The output prints the sum.

**Sample InputSample Output**

5

1 5 9 7 8

30

#include <stdio.h>

int sum(int arr[], int start, int len)

{

if(start >= len)

return 0;

return (arr[start] + sum(arr, start + 1, len));

}

**Footer**

int main()

{

int N, i, sumofarray;

scanf("%d", &N);

int arr[N];

for(i=0; i<N; i++)

{

scanf("%d", &arr[i]);

}

sumofarray = sum(arr, 0, N);

printf("%d", sumofarray);

return 0;

}

Cod

Q1.

**Math Olympiad**

The KISA Math Olympiad is held annually across all the Schools to identify and encourage the mathematical creativity of children of all grades. Samuel is a little genius in math in Grade 5 and participated in the contest.

The question bank of the contest contains 35 questions and lasts for the duration of 1 hour. Samuel was so confident in all the questions but he seemed stuck on one question. He needs your help in answering that question so that he can score full marks and get a Gold medal in the event. The question that needs your assistance is as follows:

Write a **recursive function** to find the m-th summation of first n natural numbers. m-th summation of first n natural numbers is defined as follows.

If m > 1

SUM(n, m) = SUM(SUM(n, m - 1), 1)

Else

SUM(n, 1) = Sum of first n natural numbers.

We are given m and n, we need to find SUM(n, m).

**Function Specifications:**

Use the function name, return type, and the argument type as:

**int summation(int, int)**

This recursive function should return the m-th summation of the first n natural numbers.

#include<stdio.h>

int summation(int n,int m)

{

if(m==0)

return n;

if(m!=0)

{

n=n\*(1+n)/2;

n=summation(n,m-1);

}

return n;

}

int main()

{

int n,m,i;

scanf("%d%d",&n,&m);

i=summation(n,m);

printf("%d",i);

return 0;

}

Q2.

Write a recursive function to find the sum of N natural numbers.

Note: Use a suitable recursive function.

**Input Format**

The input consists of an integer **N.**

**Output Format**

The output prints the sum of **N** numbers.

#include <stdio.h>

#include <stdlib.h>

int addNumbers(int n)

{

if(n != 0)

return n + addNumbers(n-1);

else

return n;

}

int main()

{

int num;

scanf("%d", &num);

printf("Sum = %d",addNumbers(num));

return 0;

}

Day 7

Q1. Write a recursive program to find the mean of an array.

**Input Format**

The first line of the input consists of the value of n.

The next input is the array elements.

**Output Format**

The output prints the mean value of the given array.

#include <stdio.h>

float findMean(int A[], int N)

{

if (N == 1)

return (float)A[N-1];

else

return ((float)(findMean(A, N-1)\*(N-1) +

A[N-1]) / N);

}

int main()

{

int i,N;

scanf("%d",&N);

int A[N];

for(i=0;i<N;i++) {

scanf("%d",&A[i]);

}

printf("%.2f\n", findMean(A, N));

**Footer**

return 0;

}

Q2. Write a program to find the maximum and minimum elements in an array using functions.

#include <stdio.h>

int maximum(int array[], int index, int len);

int minimum(int array[], int index, int len);

int main()

{

int N, max, min;

int i;

scanf("%d", &N);

int array[N];

for(i=0; i<N; i++)

{

scanf("%d", &array[i]);

}

max = maximum(array, 0, N);

min = minimum(array, 0, N);

printf("Minimum element in array = %d\n", min);

printf("Maximum element in array = %d\n", max);

return 0;

}

int maximum(int array[], int index, int len)

{

int max;

if(index >= len-2)

{

if(array[index] > array[index + 1])

return array[index];

else

return array[index + 1];

}

max = maximum(array, index + 1, len);

if(array[index] > max)

return array[index];

else

return max;

}

int minimum(int array[], int index, int len)

{

int min;

if(index >= len-2)

{

if(array[index] < array[index + 1])

return array[index];

else

return array[index + 1];

}

min = minimum(array, index + 1, len);

if(array[index] < min)

return array[index];

else

return min;

}

Cod

Q1.

**Entrance Test**

"Success Academy" has arranged an entrance test for High School students from rural villages. Those successful students of the test will be awarded a scholarship for their IIT/JEE preparations at Success Academy. Sunil, the co-coordinator and founder of the academy has given one problem for the first stage of the test. The problem goes like this:

Given two integers x and n, find the number of ways to express x as the sum of n-th powers of unique natural numbers. It is given that 1 <= n <= 20.

Sunil himself has not computed the solution to the problem. Write a recursive function to help him find the answer for the same to evaluate the students.

**Function Specifications:**

Use the function return type and the argument type as:

**int countWays(int,int)**

This recursive function should return the number of ways to express x as the sum of n-th powers of unique natural numbers.

#include<stdio.h>

#include<math.h>

int count(int x,int n,int num)

{

int i;

i=x-pow(num,n);

if(i==0)

{

return 1;

}

if(i<0)

{

return 0;

}

else

return count(i,n,num+1)+count(x,n,num+1);

}

int countWays(int x,int n)

{

return count(x,n,1);

}

int main()

{

int x,n;

scanf("%d%d",&x,&n);

printf("%d",countWays(x,n));

return 0;

}

Q2. Write a recursive function to count from any number n (between 1 and 9) to the number 10.

#include<stdio.h>

void count(int n)

{

if((n<10)&&(n>1))

{

printf("%d ",n);

count(n+1);

}

if(n==10)

{

printf("10");

}

}

int main()

{

int n;

scanf("%d",&n);

count(n+1);

return 0;

}

Day 8

Q1. Complete the function **fun()** with one local variable and one static variable. Print the values in both variables and decrement by 2. Call the display function twice from the main function.

#include <stdio.h>

void fun(){

//int n1 = 10;

//static int n2 = 10;

int n1;

static int n2;

scanf("%d %d",&n1,&n2);

printf("Local=%d Static=%d\n", n1, n2);

n1-=2;

n2-=2;

}

**Footer**

int main()

{

fun();

fun();

return 0;

}

1. Declare a variable **a** in two inner blocks and increase the numbers by 10 each.

#include <stdio.h>

int main( )

{

int a;

scanf("%d",&a);

{

int a;

scanf("%d",&a);

printf( "%d", a+10);

}

printf("\n%d", a+10);

}

Cod

Q1. Complete the function **display()** with one local variable and one static variable. Print the values in both variables and increment by 1. Call the display function thrice from the main function.

#include <stdio.h>

void display()

{

int n1;

static int n2;

scanf("%d%d",&n1,&n2);

printf("Local=%d Static=%d\n", n1, n2);

n1++;

n2++;

}

int main()

{

display();

display();

display();

return 0;

}

Q2. The simple measure of body fitness is the BMI or Body Mass Index. It depends only on the height L and weight W of a person. It is defined as BMI = [weight / height^2] where weight is taken in kilograms and length in meters.

Four general grades are proposed:

1. Underweight[U] - when BMI < 18.5
2. Normal weight[N] - 18.5 <= BMI < 25.0
3. Heavyweight [H] - 25.0 <= BMI < 30.0
4. Overweight [O] - above or equal to 30.0

Write a program that takes in the Weight (in Kg) and Length (in meters) of an individual and displays the grade as U, N, H, O.

**Note: Store all the variables in register memory.**

#include<stdio.h>

int main()

{

register int n;

register float m;

scanf("%d %f",&n,&m);

register float b=n/(m\*m);

if( b < 18.5)

printf("U");

else if (18.5 <= b && b< 25.0 )

printf("N");

else if(25.0 <= b && b< 30.0)

printf("H");

else

printf("O");

}

Mcqs

Day 1

Q1. All of the above.

Q2 All of the above.

Q3. II and III

Q4. File scope

Q5. pow(x,2)

**Day 2**

Q1. Passed by value

**Solution**

In C, function parameters are always passed by value. Pass-by-reference is simulated in C by explicitly passing pointer values.

Q2. Functions can return any type except array and functions

Q3. All of the mentioned

Q4. Both Library and User Defined

Q5. No limit

Day 3

Q1.main()

Q2. 1

Q3. None of the above

Q4. False

**Solution**

Since the scope of the variable declared within a function is restricted only within that function, the above statement is false.

Q5. False

**Solution**

Since the scope of the variable declared within a function is restricted only within that function, the same name can be used to declare another variable in another function.

Day 4

Q1. Compile time error

Q2. Compile time error

Q3. 0 3 0 0 0 0

Q4. func(array);

Q5. All of the mentioned

**Day 5**

Q1. 1 2 3 4 5

Q2. i value is 1 and j value is 1

Q3. i value is 1 and j value is 7

Q4 .15 5

Q5. fun2()

**Day 6**

Q1. Run time error

Q2.False

Q3. B) Stack Overflow

Q4. Loop

Q5.True

**Day 7**

Q1. Recursion uses more memory compared to iteration

Q2 . (B) return

Q3. Sum:=N+Sum(N-1)

Q4. 9

Q5. 16

**Day 8**

Q1. Both I & II are correct

Q2. Struct

Q3. In memory

Q4. (C) 7

Q5. (B) b